23108 5

UCRL-ID-126849

Fourier Analysis

J. Lawson

March 3, 1997

Lawrence Livermore National Laboratory

Work performed under the auspices of the U.S. Department of Energy by the Lawrence Livermore National Laboratory under Contract W-7405-Eng-48.

Available to DOE and DOE contractors from the
Office of Scientific and Technical Information
P.O. Box 62, Oak Ridge, TN 37831
Prices available from (615) 576-8401, FTS 626-8401

Available to the public from the
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Rd.,
Springfield, VA 22161

ucrl -

March 3, 1997

Hi Qiang,

Sorry it took so long to respond to your request for more information. It took longer than I anticipated to piece together everything that I had done. What follows is a description of my analysis.

First, the FFT that I use is described on the attached pages. Note that the scaling factor for the forward transform is 1/N.

I compute the following rms values:
rms(original data) = 64.9463 nm
rms(data*hanning) = 55.7723 nm (before renormalization)

The use of the hanning filter is accompanied by a renormalization to insure that the rms value is maintained.

I also fit to the curvature of the scan. The data corrected for focus gives the following rms values:
rms(corrected data) = 56.8835 nm
rms(corrected data*hanning) = 53.2179 nm (before renormalization)

The PSD is shown for various data. The PSD is calculated as:
PSD = | FFT(y) | ^2 * xl
where xl is the length of the x axis, 45.9952.

I did find an error in the plot that you were sent. If kx is the frequency axis, i.e., values from (0,Nyquist), then kx(1,Nyquist) is plotted versus PSD(0,Nyquist). This error is corrected in the attached plots. The plot you have appears to be the PSD of the original data with no hanning applied.

The removal of the quadratic term appear to have a negligible effect on the PSD. It changes only the first couple of terms (which lie outside of the data valid range). The removal of the center feature has a much stronger effect.

Janice JKL

FFT

The FFT function returns a result equal to the complex, discrete Fourier transform of *Array*. The result of this function is a single– or double–precision complex array.

The discrete Fourier transform, $F(u)$, of an N–element, one–dimensional function, $f(x)$, is defined as:

$$F(u) = \frac{1}{N}\sum_{x=0}^{N-1} f(x)\exp[-j2\pi ux/N]$$

And the inverse transform, (*Direction* > 0), is defined as:

$$f(x) = \sum_{u=0}^{N-1} F(u)\exp[j2\pi ux/N]$$

If the keyword OVERWRITE is set, the transform is performed in–place, and the result overwrites the original contents of the array.

The result returned by FFT is a complex array that has the same dimensions as the input array. The output array is ordered in the same manner as almost all discrete Fourier transforms. Element 0 contains the zero frequency component, F0. F1 contains the smallest non–zero positive frequency, which is equal to $1/(NT)$ where N is the number of elements in the respective dimension and T is the sampling interval. F2 corresponds to a frequency of $2/(NT)$. FN/2 contains the component corresponding to the Nyquist frequency, $1/(2T)$, which is the highest frequency that can be sampled.

Negative frequencies are stored in reverse order from FN–1, FN–2, ..., FN/2+1, corresponding to frequencies of $-1/NT$, $-2/NT$, and $-(N/2 - 1)/NT$, respectively.

Calling Sequence

Result = FFT(*Array [, Direction]*)

Arguments

Array

The array to which the Fast Fourier Transform should be applied. If *Array* is not of complex type, it is converted to complex type. The dimensions of the result are identical to those of *Array*. The size of each dimension may be any integer value and does not necessarily have to be an integer power of 2, although powers of 2 are certainly the most efficient.

Direction

Direction is a scalar indicating the direction of the transform, which is negative by convention for the forward transform, and positive for the inverse transform. If *Direction* is not specified, the forward transform is performed.

A normalization factor of $1/N$, where N is the number of points, is applied during the forward transform.

Note: When transforming from a real vector to complex and back, it is slightly faster to set *Direction* to 1 in the real to complex FFT.

Note also that the value of *Direction* is ignored if the INVERSE keyword is set.

Keywords

DOUBLE

Set this keyword to a value other than zero to force the computation to be done in double–precision arithmetic, and to give a result of double–precision complex type. If DOUBLE is set equal to zero, computation is done in single–precision arithmetic and the result is single–precision complex. If DOUBLE is not specified, the data type of the result will match the data type of *Array*.

INVERSE

Set this keyword to perform an inverse transform. Setting this keyword is equivalent to setting the *Direction* argument to a positive value. Note, however, that setting INVERSE results in an inverse transform even if *Direction* is specified as negative.

OVERWRITE

If this keyword is set, and the *Array* parameter is a variable of complex type, the transform is done "in–place". The result overwrites the previous contents of the variable. For example, to perform a forward, in–place FFT on the variable a:

```
a = FFT(a, -1, /OVERWRITE)
```

Running Time

For a one–dimensional FFT, running time is roughly proportional to the total number of points in *Array* times the sum of its prime factors. Let N be the total number of elements in *Array*, and decompose N into its prime factors:

$$N = 2^{K_2} \cdot 3^{K_3} \cdot 5^{K_5} \ldots$$

Running time is proportional to:

$$T_0 + N(T_1 + 2K_2T_2 + T_3(3K_3 + 5K_5 + \ldots))$$

where *T3 ~ 4T2*. For example, the running time of a 263 point FFT is approximately 10 times longer than that of a 264 point FFT, even though there are fewer points. The sum of the prime factors of 263 is 264 (1 + 263), while the sum of the prime factors of 264 is 20 (2 + 2 + 2 + 3 + 11).

Example

Display the log of the power spectrum of a 100–element index array by entering:

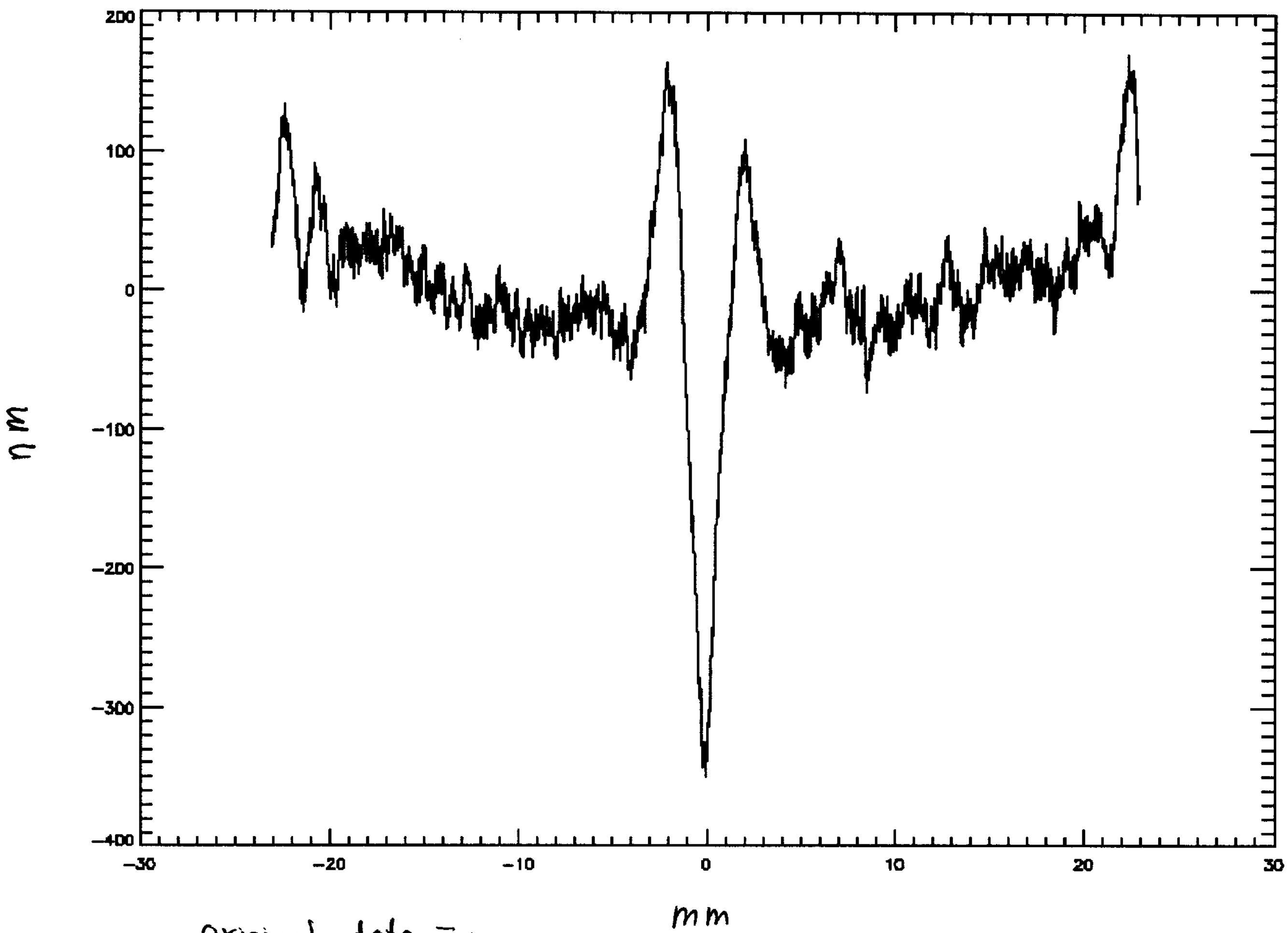
```
PLOT, /YLOG, ABS(FFT(FINDGEN(100), -1))
```

As a more complex example, display the power spectrum of a 100–element vector sampled at a rate of 0.1 seconds per point. Show the 0 frequency component at the center of the plot and label the abscissa with frequency:

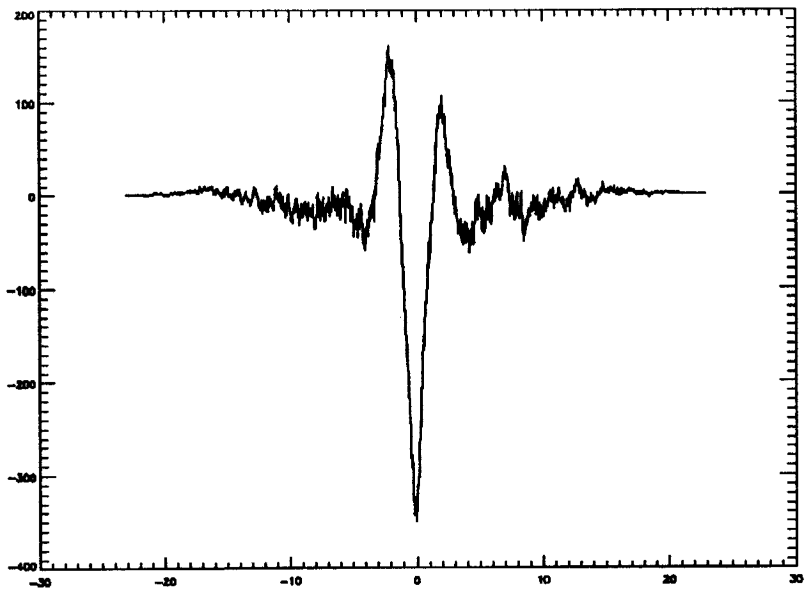
`N =100`	*Define the number of points.*
`T = 0.1`	*Define the interval.*
`N21 = N/2 + 1`	*Midpoint+1 is the most negative frequency subscript.*
`F = INDGEN(N)`	*The array of subscripts.*
`F(N21) = N21 -N + FINDGEN(N21-2)`	*Insert negative frequencies in elements F(N/2 +1), ..., F(N–1).*
`F = F/(N*T)`	*Compute T0 frequency.*
`PLOT, /YLOG, SHIFT(F, -N21), SHIFT(ABS(FFT(Y, -1)), -N21)`	*Shift so that the most negative frequency is plotted first.*

See Also

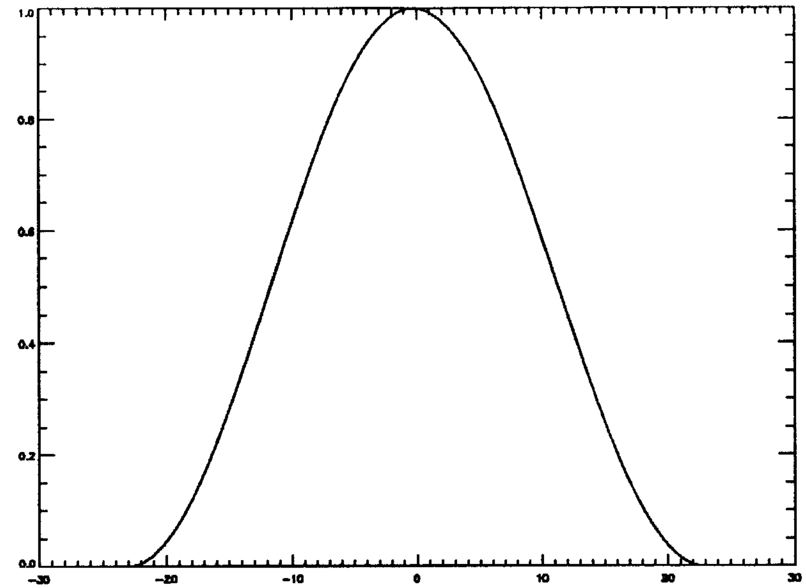
HILBERT



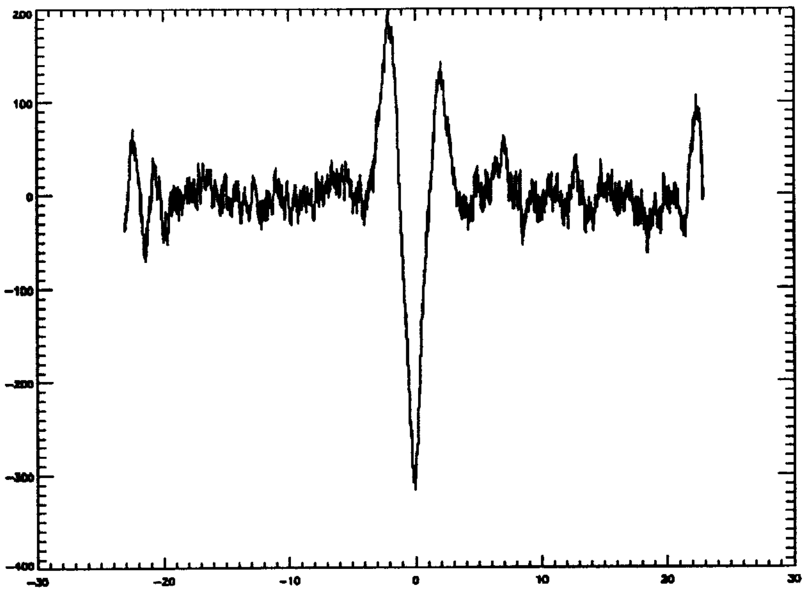
original data ≡ y



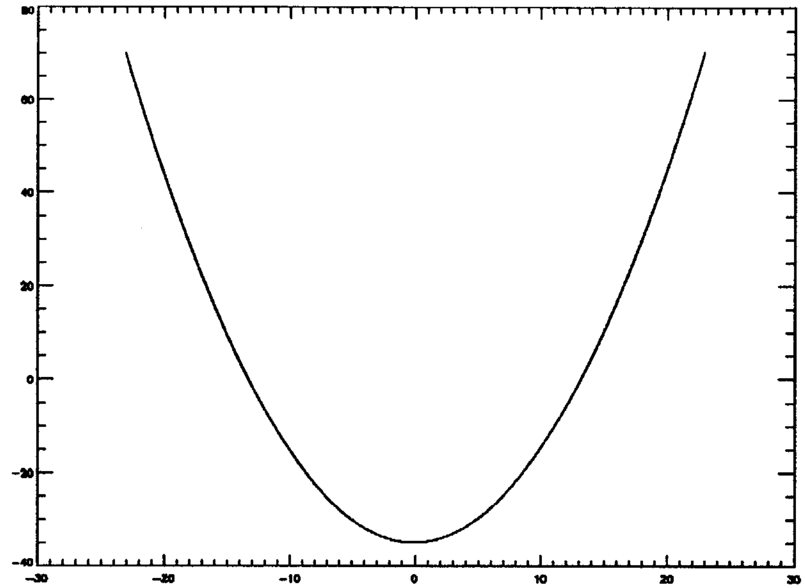
y*hanning



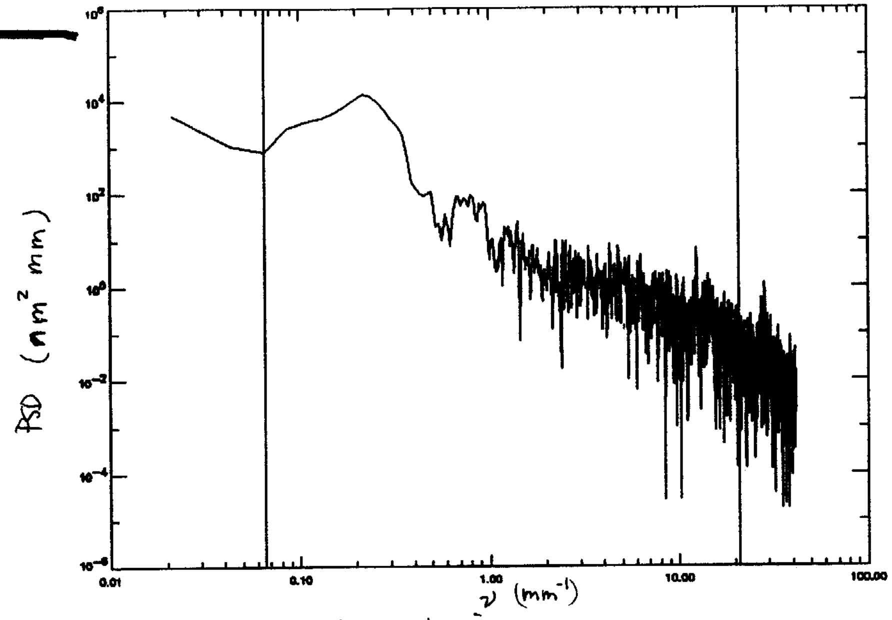
hanning



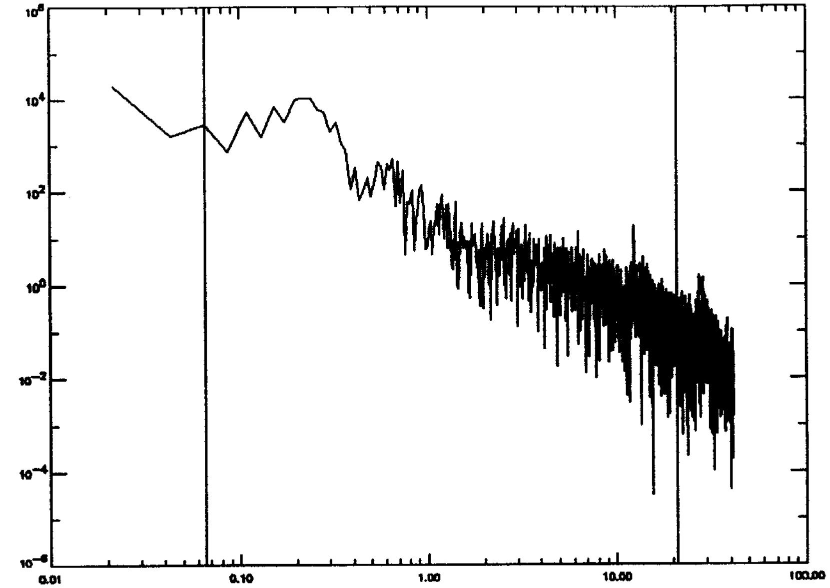
y corrected for curvature



quadratic fit to data

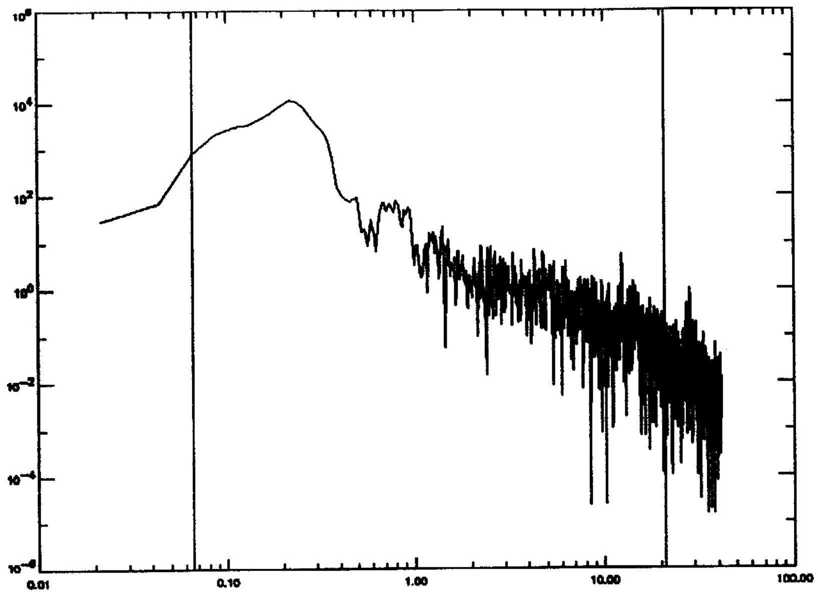


PSD of y*hanning

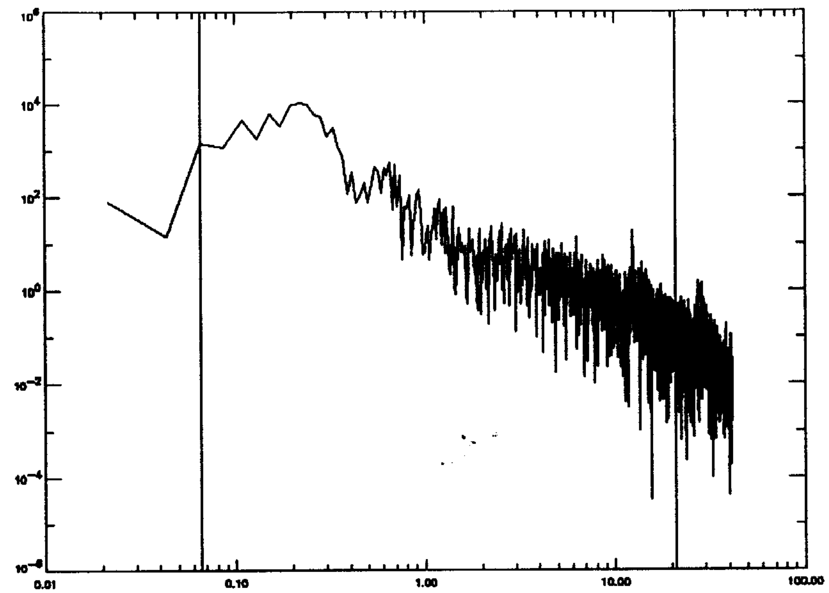


PSD = |FFT(y)|² * XL

8

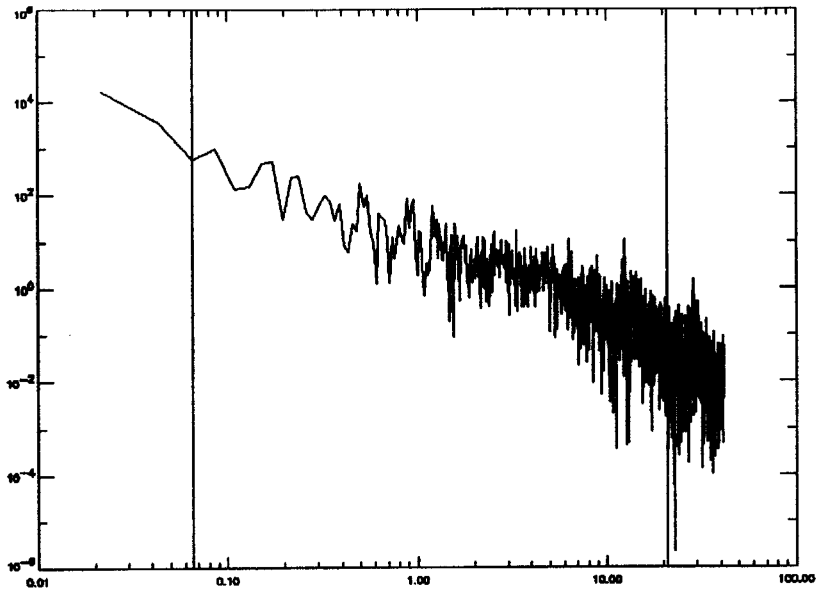


PSD of ycorrected * hanning

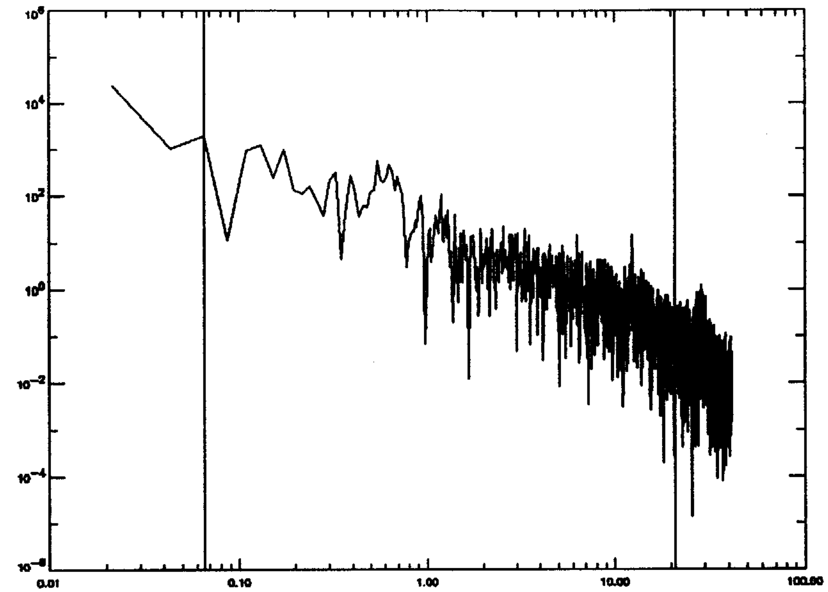


PSD of ycorrected

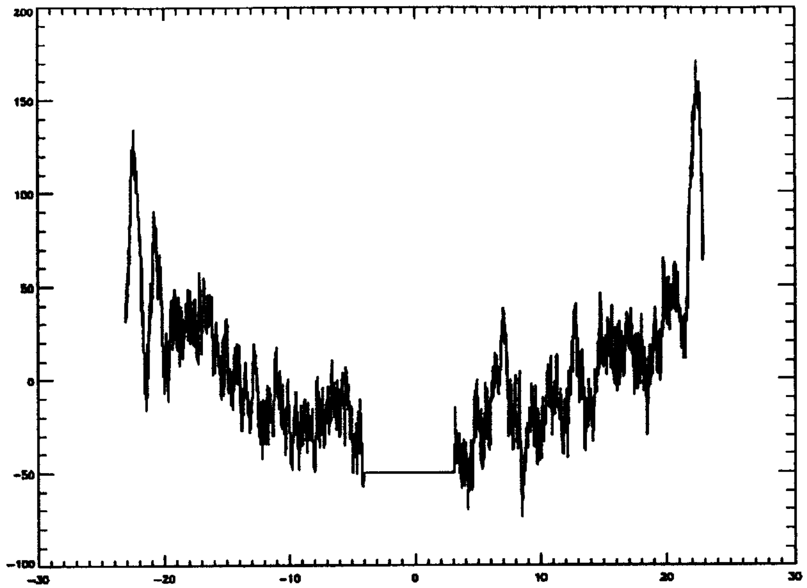
6



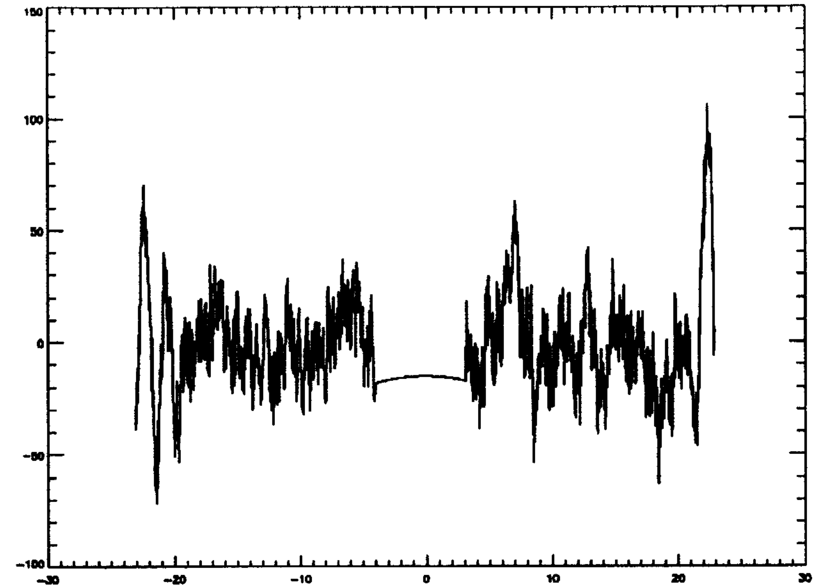
PSD of yfix * hanning



PSD of yfix

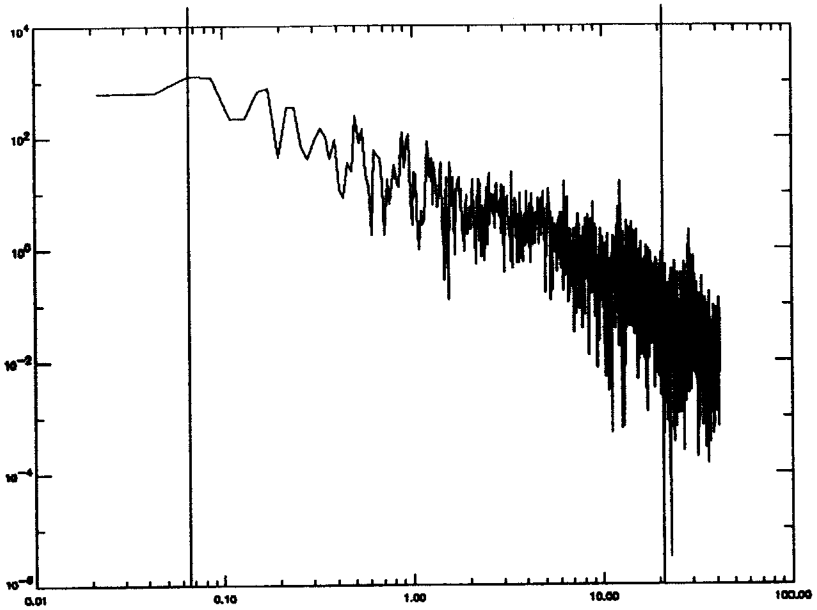


yfix = center feature removed

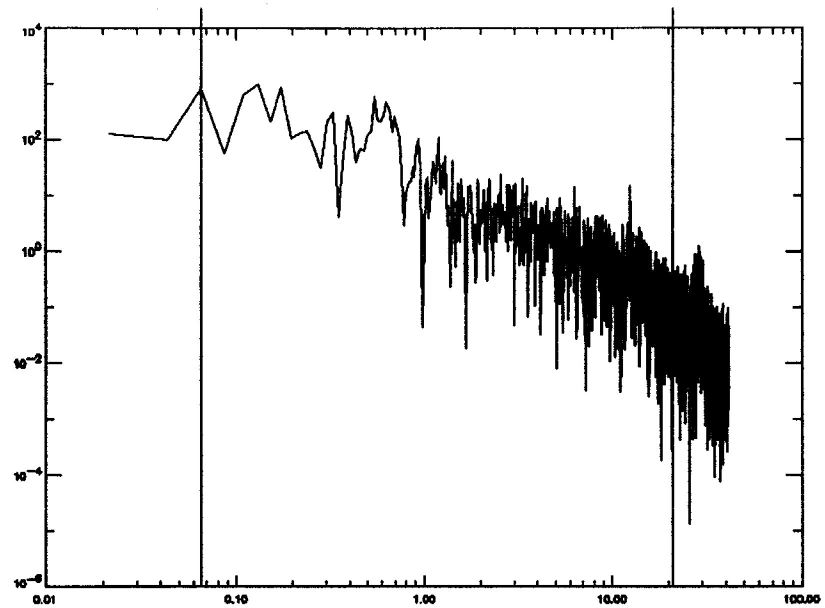


yfixc = yfix - quadratic fit

//



PSD of yfixc * hanning



PSD of yfixc